IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of	
John Wong	Group Art Unit: 2168
Application No.: 09/810,246	Examiner: Haythim J. Alaubaidi
Filed: March 19, 2001	Confirmation No.: 8377
For: MIRROR FILE SYSTEM	

BRIEF FOR APPELLANT

Mail Stop APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This appeal is from the decision of the Primary Examiner dated February 10, 2006, finally rejecting claims 1-16, which are reproduced as an Appendix to this brief.

A check covering the requisite fee under 37 CFR 41.20(b)(2) was submitted with the Brief filed October 18, 2004. The Commissioner is hereby authorized to charge any additional fees that may be required by this paper, and to credit any overpayment, to Deposit Account No. 02-4800.

Table of Contents

Page

I. Real Party in Interest

The subject application is assigned to Twin Peaks Software, Inc., a California corporation.

II. Related Appeals and Interferences

There are no other known prior or pending appeals, interferences or judicial proceedings which may be related to, directly affect, or be directly affected by, or have a bearing on, the Board's decision in this appeal.

III. Status of Claims

The application contains claims 1-16, all of which are currently pending, and stand finally rejected. This appeal is directed to all rejected claims.

IV. Status of Amendments

There were no amendments filed subsequent to the final Office Action.

V. Summary of Claimed Subject Matter

A. Overview

The claimed subject matter is directed to file systems for computers, and more particularly to a virtual file system that links two or more physical file systems together, and mirrors between them in real time. The structure and operation of the invention can be understood with reference to Figures 3 and 4 of the application. Figure 3 illustrates two physical file systems A and B. As is conventional, each file system contains a number of directories and files that are organized in a hierarchical manner. For example, file system A contains a root directory /A and two subdirectories b and i at a first level under the root directory. Further directories and/or files are arranged at lower levels of the hierarchy. In the example of Figures 3 and 4, a portion of the structure of file system A is to be linked to file system B. This linked structure comprises the directory b and each of the directories or files c-h that are nested under directory b. This structure, labeled 220, is to be linked to

the structure 221 that includes the nested directories and/or files y, z and v in file system B. (Page 19, lines 3-6).

Referring to Figure 4, the directory y of file system B is mounted onto the directory b of file system A. As a result, it can be seen in Figure 4 that the directory b now contains, in addition to subdirectories c-h, the subdirectories z and v that were contained in directory y of file system B. In addition, the directory b of file system A is mounted onto the directory y of file system B. As illustrated in Figure 4, directory y of file system B therefore contains subdirectories c-h, as well as subdirectories z and v. (Page 19, lines 10-15).

A virtual file system 203 is created in conjunction with these mounting operations. This virtual file system has a data structure that contains all of the elements of the structure 220 of file system A and the structure 221 of file system B. Thus, it has a root directory b/y 231, and subdirectories or files c-h, z and v. (Page 19, line 19 to page 20, line 11).

Every element of a file system, namely each file and each directory, has a data structure known as a *vnode* that contains information and the operations that can be performed on the file or directory. In the virtual file system disclosed in the application, every file or directory in the virtual file system 203 has a super *vnode* structure that is called an *mnode*. This *mnode* contains a *vnode* structure and two *vnode* pointers. The *vnode* structure comprises the *vnode* for the given file or directory within the virtual file system 203, and the two *vnode* pointers respectively point to the actual *vnode* of the corresponding file or directory within the two physical file systems A and B. The two *vnode* pointers are represented in Figure 4 by the small black squares within the directories 231 to 233, 236 and 238, with arrows that point to their corresponding structures in the file systems A and B. (Page 21, lines 5-16; page 21, line 29 to page 22, line 7).

In operation, when a request is made to access a file or directory under the b directory of file system A or the y directory of file system B, the file system detects that these directories have the virtual file system 203 mounted on them. As a result, all file access requests are directed to the *vnode* for the virtual directory b/y of the virtual file system 203. Using the *mnode* data structure, the system obtains the *vnodes* for both the directory b in file system A and directory y in file system B. The requested operation, e.g. a write operation, is then sent to both of the *vnodes*. As a result, real time mirroring is

effected between the corresponding pairs of elements, i.e. files or directories, in the two physical file systems A and B. (Page 22, line 8 to page 23, line 32).

A mirror file system of this type can be configured in a number of different ways, as depicted in Figures 5-10 of the application. Exemplary Figure 8 illustrates a client system 700 that uses a mirror file system Z to access two physical file systems A and B, for example on different servers 710 and 720. In this configuration, file system A and file system B are exported to the client, where they are mounted by the virtual file system Z. In this configuration, operations performed by the client upon elements stored in the file system of either server are automatically mirrored to the file system of the other server, in a real-time fashion. (Page 29, lines 1-22).

B. The Claims

The application contains three independent claims, namely claims 1, 4 and 11. Claim 1 recites a virtual file system having means for mounting components of each of two physical file systems in a single directory. In the disclosed embodiment, this means is implemented by the mirror files system interface 22 running on a suitable computer (page 15, line 19 to page 16, line 11; page 17, line 12 to page 19, line 2). The second element of claim 1 is a virtual file system data structure containing elements which correspond to each of the mounted components. This data structure is shown in Figure 4, containing elements 231-239 that respectively correspond to the components b-h, y, z and v of the two files systems A and B (page 19, line 22 to page 20, line 2). Each of these elements has an application interface data structure with two associated pointers that respectively point to application interface data structures of a corresponding component in each of the two physical file systems. Referring to element 231, it has an *mnode* with two pointers that respectively point to the *vnode* structure of directory b in file system A, and directory y in file system B (page 22, lines 2-7).

Claim 4 recites a method that includes the first step of mounting components of each of two file systems in a single directory, such that a copy of each component is stored in each of the two physical file systems (page 20, lines 5-11; Figure 4, elements 204 and 205). The second step of claim 4 is receiving a request to perform a write operation on one of the components (page 22, lines 14-16). The final step of claim 4 is performing the write

operation on both copies of the component in the two file systems, respectively, in real time (page 23, lines 28-32).

Claim 11 recites a mirrored file system comprising first and second servers each having a local file system and a physical storage device (Figure 8, servers 710 and 720 with local file systems 711 and 721, and physical storage 712 and 722). The system further includes a client device (700) having a virtual file system (701) which mounts imported file systems from the two servers (702 and 703) to provide a single point of access for components stored in each of the first and second local file systems (page 29, lines 1-22).

VI. Grounds of Rejection to Be Reviewed on Appeal

The final Office Action sets forth three grounds of rejection that are to be reviewed on this appeal:

1. Are claims 1-5 and 7-10 unpatentable under 35 U.S.C. §103 when the Duso patent (US 6,625,750) is considered in view of the Enoki patent (US 5,873,085)?

2. Is claim 6 unpatentable under 35 U.S.C. §103 when the Duso and Enoki patents are considered in view of the Mukherjee patent (US 6,466,978)?

3. Are claims 11-16[1] unpatentable under 35 U.S.C. §103 when the Enoki patent is considered in view of the Soltis patent (US 6,697,846)?

VII. Argument

A. Claims 1-5 and 7-10

Claims 1-5 and 7-10 were rejected under 35 U.S.C. 103(a) as being unpatentable over the Duso et al patent (US 6,625,750) in view of the Enoki et al patent (US 5,873,085). The criteria for a supportable rejection under 35 U.S.C. § 103 are set forth in MPEP §2143. One of these criteria is that "the prior art reference (or references when combined) must teach or suggest all of the claim limitations." As illustrated below, the rejections of the claims do not meet this requirement.

[1] Although the statement of rejection at the top of page 9 of the final Office Action only identifies claims 11-14, the subsequent discussion of the rejection also includes claims 15 and 16. For purposes of this Brief, Appellant is assuming the examiner intended to reject each of claims 11-16 under this ground of rejection.

1. Claims 1-3 and 7

The final Office Action asserts that the Duso patent discloses all of the subject matter recited in claims 1-3, 7 and 9, with the exception of elements in a virtual file system "having an application i nterface data structure with two associated pointers that respectively point to application interface data structures of a corresponding component in each of ... two physical file systems" as recited in claim 1. Before addressing this distinction, Appellant notes that there are additional differences between the claimed subject matter and the Duso patent.

Specifically, claim 1 recites "a virtual file system data structure containin g elements which respectively correspond to each of the mounted components." In connection with this subject matter, the Office Action refers to the Duso patent at Column 9, lines 47-65; and Column 10, lines 48-50. The Duso patent describes how a single file system, UFS or CMFS, interfaces with the operating system's Virtual File System (VFS) and Network File System (NFS). The term VFS, as used in the patent, refers to an industry-standard *back-end file system switch* (Column 9, lines 55-57; Column 10, lines 47-49). This disclosure does not suggest the claimed subject matter, namely a data structure of the virtual file system that contains elements that respectively correspond to *each* of the mounted (e.g. UFS and NFS file systems) components. The Duso patent does not suggest that the industry-standard back-end file system switch contains elements that correspond to those in each of two mounted physical file systems. Rather, as its name implies, it functions as a *switch* to direct read and write requests to the appropriate one of the two physical file systems. The Office Action does not identify where a *data structure* of the type recited in claim 1 is disclosed in the Duso patent.

In connection with the acknowledged difference noted above, i.e. an application interface data structure with two associated pointers, the Office Action refers to the Enoki patent at Figures 3, 8, 11, 15-18, 30, 32, 37a-37b and 46a-46b, as well as Column 3, lines 9-16 and the Abstract. Each of the cited figures depicts an example of a management table that provides a mapping between a virtual file identifier and a real file identifier. The final Office Action does not explain how these figures can be interpreted to suggest the claimed subject matter. The Enoki patent discloses a virtual file management apparatus that is concerned with directing a file request from a terminal to a real file stored on a file server on a network. All of the figures and text pertain to mapping between a *single* virtual file

identifier and a *single* real file identifier on one server. The Enoki patent does not disclose how a write request to a virtual file identifier from a terminal can be directed to two real files stored on two different servers. As such, it does not disclose that each element in a file system data structure has an application interface data structure with "*two* associated pointers that *respectively* point to application interface data structures of a corresponding component in *each* of *two* physical file systems" as recited in claim 1. Consequently, any possible application of Enoki's teachings t o the system of the Duso patent would not result in the claimed subject matter.

2. Claim 8

Claim 8 recites that the single directory functions as a single mount point for access to the components of either of the two physical file systems. In connection with this claimed subject matter, the final Office Action refers to the Enoki patent at Figure 30, and column 1, lines 30-37.

Appellant is unable to determine how these portions of the Enoki patent are being interpreted to suggest the claimed subject matter. This patent does not disclose that two physical file systems are mounted in a single directory. Referring to Figure 30, even if one considers the file systems 3004a and 3004b to be two distinct physical file systems, there is no disclosure that these two file systems are mounted in a single directory. As such, it is not apparent how the patent can teach that a single directory can function as a single mount point for access to the components of either of two physical file systems.

The Office Action merely cites to the two portions of the Enoki patent, without explaining the basis for the rejection of claim 8. It does not provide sufficient evidence to support the rejection.

3. Claim 9

Claim 9 depends from claim 1, and recites that the mounted components of each physical file system are replicated in the other physical file system. The final Office Action does not address this claimed subject matter, and specifically does not identify where it is taught in the references.

As noted previously, the final Office Action asserts that all of the subject matter of claims 1-3, 7 and 9 is disclosed in the Duso patent, with the exception of one feature recited in claim 1. Presumably, therefore, the Duso patent is being relied upon to teach the

replication feature of claim 9. However, it is not disclosed in the passage cited in the Office Action, i.e. column 10, lines 39-48. At best, this passage states that the Continuous Media File System (CMFS) may be mounted on a directory within the UFS hierarchy. It does not, however, state that the components of the CMFS are replicated in the UFS, or vice versa.

The final Office Action does not provide any support for the rejection of claim 9, since it fails to identify where the subject matter of that claim is disclosed in either of the references used to reject the claim.

4. Claims 4 and 10

In rejecting claims 4 and 10, the Office Action states that the Duso reference discloses all of the claimed subject matter, with the exception of the step of receiving a request to perform a write operation on one of the components, and performing the write operation on both copies of the component in two physical file systems. To this end, it refers to the Enoki patent at Column 15, lines 17-26 and Column 28, lines 9-27.

As discussed above, there are additional differences between the disclosure of the Duso patent and the claimed subject matter. Specifically, the Duso patent does not disclose that a copy of each mounted component of two physical file systems is stored in each of the two physical file systems, as recited in claim 4. See the previous discussion of claim 9. For at least this reason, therefore, the Office Action has not shown that the references teach every element of the rejected claims.

Furthermore, Enoki's teaching pertains to how a write request from a terminal is to be redirected to a *single* server, where the real file is stored, by means of a one-to-one mapping of the management table. The cited portions of the patent do not disclose that a write request is performed on *both* copies of a component in *two* physical file systems, respectively.

For this additional reason, therefore, the Office Action does not establish that all of the features recited in the rejected claims are disclosed in the references.

5. Claim 5

Claim 5 recites that the request designates a component on which the write operation is to be performed by means of a path name that is common to both of two physical file systems. In rejecting this claim, the Office Action cites the Enoki patent at column 3, lines

9-30, and column 15, lines 17-26, with specific reference to a virtual file identifier and a server name.

Neither of these passages discloses a path name in a write request that is common to both of *two* physical file systems. Since the Enoki patent is directed to operations on a single server having a single file system, it cannot be interpreted to suggest the additional elements of the invention recited in claim 5, which pertain to features that are common to two physical file systems, i.e. path names.

B. Claim 6

Claim 6 recites that the step of performing a write operation includes acquiring a lock for each copy of a component, and inhibiting the write operation until both locks can be acquired. This claim was rejected under 35 U.S.C. 103(a) as being unpatentable over the Duso and Enoki patents in further in view of the Mukherjee patent. The Mukherjee patent was relied upon for its alleged teaching of acquiring a lock for each copy of a component, and inhibiting write operations until both locks can be acquired, with reference to Column 17, lines 56 through Column 18, line 15. However, the Mukherjee patent only discloses the locking of two distinct status tables within a system. It does not disclose how the inhibit process would work if one table is locked and the other table cannot be locked. More significantly, there is no disclosure suggesting the use of two locks that are respectively associated with each of two *copies* of a component in respective file systems.

C. Claims 11-16

1. Claims 11-13

Claims 11-16 were rejected under 35 U.S.C. 103(a) as being unpatentable over the Enoki patent in view of the Soltis patent (US 6,697,846). Claim 11 recites a mirrored file system that comprises a first server having a first local file system and a first physical storage device, and a second server having a second local file system and a second physical device. The Office Action states that the Enoki patent discloses: a first server (Figure 30, Element 3001a) having a first local file system (Figure 30, Element 3004a) and a first physical storage device; and a second server (Figure 30, Element 3001b) having a second local file system (Figure 30, Element 3004b) and a first physical storage device.

Claim 11 goes on to recite a client device having a virtual file system which mounts an imported file system from the first server and an imported file system from the second server, to provide a single point of access for components stored in each of the first and second local file systems. The Office Action alleges that the first server and second server provide a single point of access for components stored in each of the first and second local file systems, with reference to Figure 30 and its corresponding text.

Enoki's Figure 30 shows that each server has a file system. However, the file systems in that figure, located on two separate servers, have no relationship with one another. Of particular significance, they are not mounted under a single directory anywhere, whether on a server or a client (terminal), so as to provide a single access point for components stored in each of two local file systems.

The Office Action acknowledges that the Enoki reference does not disclose a client system having a virtual file system which mounts an imported file system. To this end, therefore, it refers to the Soltis patent at Figure 2, Element 122, Figure 4, and Column 9, line 65 through Column 10, line 12 as disclosing such subject matter.

The Virtual File System 122 in Figure 2 of the Soltis patent is a virtual file system interface (VFS, Column 5, line 29-33; Column 8, lines 17-20). The depiction of client 106 in Figure 2 is used to describe how the Shared File System implementation and MFS (Meta-data File System) Client implementation use the Virtual File System interface 122. It does not disclose how the Virtual File System interface mounts two file systems. A virtual file system *interface*, by its very nature, cannot mount any file system.

Thus, neither the Enoki patent nor the Soltis patent discloses a client device having a virtual file system, particularly, one that mounts an imported file system from each of two different servers. As such, any possible combination of their teachings would not result in the subject matter recited in claim 11.

Claim 12 recites that the components of each of the two imported file system are mounted to a single directory. Since the references do not disclose that two local file systems are imported into a client device, they cannot be interpreted to disclose this additional feature recited in claim 12.

Claim 13 recites that the virtual file system in the client contains elements that each have an application interface data structure with two pointers that respectively point to application interface data structures of a corresponding component in the two imported file

systems. The rejection of claim 13 relies upon the same portions of the Enoki patent that were discussed previously in connection with the rejection of claim 1. As pointed out in that discussion, the Enoki patent does not disclose a data structure of a type recited in the claim.

2. Claims 14-16

Claim 14 recites that each of the first and second *servers* includes a virtual file system that mounts components of that server's local file system *and* components of the other server's local file system in a single directory. Claim 15 recites that it is these virtual file systems that are imported into the client. The final Office Action does not address the subject matter of either of these claims, except to make the conclusory statement that it is disclosed in the Enoki patent. Neither reference contains any disclosure of a server having a virtual file system that mounts components of its own local file system and the file system of another server, nor the importation of such virtual file systems into a client device. The final Office Action does not provide any support for the rejection of claims 14 and 15.

Claim 16 corresponds to claim 13, and recites the application interface data structure of the virtual file system in each server. For the reasons presented previously in connection with claims 1 and 12, this subject matter is not disclosed in the Enoki patent.

VIII. Conclusion

In summary, the rejections are based upon conclusory statements that the features recited in the claims are disclosed in the references. However, the passages that are cited in the rejections do not support these conclusions. In response to Appellant's arguments traversing the rejections, the final Office Action merely repeats the citations without offering any explanation how they are being interpreted in light of the claim language. The final Office Action does not provide the necessary showing that the references, when combined, disclose all the claimed subject matter, as required in MPEP §2143.

The rejections of the claims are not properly founded in the statute, and should be reversed.

Respectfully submitted,

BUCHANAN INGERSOLL & ROONEY PC

Date: December 11, 2006

By: _____

James A. LaBarre
Registration No. 28,632

P.O. Box 1404
Alexandria, Virginia 22313-1404
(703) 836-6620

IX. APPENDIX A - The Appealed Claims

1. A virtual file system which provides mirroring and linking of two physical file systems, comprising:

means for mounting components of each of said two physical file systems in a single directory; and

a virtual file system data structure containing elements which respectively correspond to each of the mounted components, each of said elements having an application interface data structure with two associated pointers that respectively point to application interface data structures of a corresponding component in each of said two physical file systems.

2. The virtual file system of claim 1, wherein said application interface data structures correspond to a vnode structure.

3. The virtual file system of claim 1, wherein said components comprise directories and files.

4. A method for sharing files in a computer system, comprising the steps of:

mounting components of each of two physical file systems in a single directory, such that a copy of each component is stored in each of said two physical file systems;

receiving a request to perform a write operation on one of said components; and

performing said write operation on both copies of said one component in said two physical file systems, respectively, in real time in response to said request.

5. The method of claim 4 wherein said request designates said one component, on which the write operation is to be performed, by means of a path name that is common to both of said physical file systems.

6. The method of claim 4 wherein the steps of performing said write operation includes the steps of acquiring a lock for each copy of said one component, and inhibiting said write operation until both locks can be acquired.

7. The virtual file system of claim 1, wherein said mounting means mounts a directory of one of said physical file systems to a directory of the other physical file system.

8. The virtual file system of claim 1 wherein said single directory functions as a single mount point for access to the components of either of said two physical file systems.

9. The virtual file system of claim 1 wherein the mounted components of each physical file system are replicated in the other physical file system.

10. The method of claim 4 wherein said mounting step comprises mounting a directory of one of said physical file systems to a directory of the other physical file system.

11. A mirrored file system, comprising:

a first server having a first local file system and a first physical storage device associated therewith;

a second server having a second local file system and a second physical storage device associated therewith; and

a client device having a virtual file system which mounts an imported file system from said first server and an imported file system from said second server to provide a single point of access for components stored in each of said first and second local file systems.

12. The mirrored file system of clam 11, wherein said first local file system and said second local file system are each imported into said client device, and said virtual file system mounts components of each of said two imported file systems to a single directory.

13. The mirrored file system of claim 12 wherein said virtual file system contains elements which respectively correspond to each of the mounted components, each of said elements having an application interface data structure with two associated pointers that respectively point to application interface data structures of a corresponding component in each of said two imported file systems.

14. The mirrored file system of claim 11 wherein each of said first and second servers includes a virtual file system that mounts components of said server's local file system and components of the other server's local file system in a single directory.

15. The mirrored file system of claim 14 wherein the file systems that are imported into said client device comprise the virtual file systems of said first and second servers.

16. The mirrored file system of claim 14 wherein the virtual file system in each server contains elements which respectively correspond to each of the mounted components, each of said elements having an application interface data structure with two associated pointers that respectively point to application interface data structures of a corresponding component in each of said two local file systems.

X. APPENDIX B - Evidence

(none)

XI. APPENDIX C – Related Proceedings

(none)